AMENDMENTS TO THE CLAIMS

1. (Previously Presented) A method of configuring a server to provide at least one composite user interface to a plurality of source applications, the composite user interface comprising a plurality of user interface elements provided by the source applications, the method comprising:

combining a subset of the plurality of user interface elements from the source applications into composite user interface data;

providing the composite user interface data to a user computer system for display as the composite user interface by the user computer system, wherein the composite user interface causes the user computer system to display the plurality of user interface elements provided by the source applications;

receiving a user request from the user computer system relevant to at least one of the source applications;

processing a model representing said composite user interface to generate rules for communication between said composite user interface and the source applications; and

generating one or more source requests to each relevant source application that represents the user request.

2. (Currently Amended) A method according to claim 1, wherein said model comprises a model of at least part of a user interface provided by each source application and a model of relationships between the at least part of the user interface provided by each source application and the composite user interface.

3. (Previously Presented) A method according to claim 1, further comprising:

storing said rules within a hierarchical data structure comprising a plurality of entities.

4. (Original) A method according to claim 3, further comprising:

storing within said hierarchical data structure an entity representing the composite user interface; and

associating with said entity a data group providing configuration data for the composite user interface.

5. (Original) A method according to claim 4, further comprising:

storing within said hierarchical data structure a plurality of service entities representing processing modules which are together adapted to process user requests input to said composite user interface to produce one or more requests to at least one source application.

6. (Original) A method according to claim 5, wherein at least some of said service entities have an associated data group storing configuration data.

7. (Previously Presented) A method according to claim 6, wherein one of said service entities is an aggregation service entity representing an aggregation service configured to generate the one or more source application requests from the user request.

8. (Previously Presented) A method according to claim 7, wherein said aggregation service entity comprises:

a child entity representing the composite user interface; and said child entity has at least one child entity representing one of the source applications.

9. (Previously Presented) A method according to claim 3, wherein said rules are generated using a plurality of writers each writer being associated with an entity in said hierarchical data structure, and being adapted to write data to a data group associated with the respective entity.

10. (Original) A method according to claim 9, wherein processing said model comprises:

selecting one or more objects within said model;

determining one or more writers to be invoked to write data from the or

each object to said hierarchical data structure; and

invoking the or each writer to write data to said hierarchical data structure.

11. (Original) A method according to claim 10, further comprising:

determining from said at least one writer at least one further object within said model, and processing said further object.

12. (Previously Presented) A method according to claim 10, further comprising:

identifying a further writer configured to identify an entity within said hierarchical data structure to which data is to be written.

13. (Original) A method according to claim 12, wherein said identifying an entity comprises:

attempting to locate an entity within said hierarchical data structure to which data should be written; and

if said attempt is unsuccessful, creating an appropriate entity.

14. (Previously Presented) A method according to claim 9, wherein each writer is a writer object which is an instance of a respective Java writer class.

15. (Original) A method according to claim 14, wherein each writer class has a corresponding writer factory class.

16. (Original) A method according to claim 15, further comprising:

registering each writer factory class with a writer lookup object;

providing details of the or each object to be processed to said writer lookup object; and

identifying one or more factory classes which should be used to create writer objects.

17. (Currently Amended) A method of generating model data representing a model of a composite user interface comprising a plurality of user interface elements provided by ~~at least one~~ a plurality of source ~~application~~ applications, the method comprising:

modelling at least part of a user interface provided by ~~the or~~ each of the source ~~application~~ applications; and

modelling relationships between the at least part of the user ~~interface~~ interfaces provided by the source ~~application~~ applications and the composite user interface.

18. (Original) A method according to claim 17, wherein the model is adapted for use in generating a composite application.

19. (Previously Presented) A method according to claim 17, wherein modelling at least part of the user interface provided by the or each source application comprises:

defining a plurality of source flow items each comprising a specified source user interface page provided by a source application; and

defining relationships between said plurality of source flow items.

20. (Original) A method according to claim 19, wherein modelling at least part of the user interface provided by the or each source application further comprises:

defining at least one page element within each specified source user interface page.

21. (Previously Presented) A method according to claim 19, wherein modelling at least part of the user interface provided by the or each source application further comprises:

defining at least one flow control condition;

associating a flow control condition with at least one of said plurality of source flow items.

22. (Previously Presented) A method according to claim 19, wherein modelling at least part of the user interface provided by the or each source application further comprises:

defining request parameters used to obtain each specified source user interface page.

23. (Previously Presented) A method according to claim 19, wherein modelling at least part of the user interface provided by the or each source application further comprises:

defining at least one rule for each specified source user interface page which can be applied to enable recognition of the associated specified source user interface page.

24. (Original) A method according to claim 23, wherein the or each rule is specified using a regular expression, or a path expression.

25. (Previously Presented) A method according to claim 17, wherein modelling at least part of the user interface provided by the or each source application further comprises:

creating a plurality of objects which are instances of classes defined in an object oriented programming language.

26. (Previously Presented) A method according to claim 17, wherein modelling relationships between the at least part of the user interface provided by the or each source application and the composite user interface comprises:

combining at least part of a plurality of source application models.

27. (Previously Presented) A method according to claim 17, further comprising:

defining a plurality of composite flow items each comprising a specified user interface page; and

defining relationships between said plurality of composite flow items.

28. (Previously Presented) A method according to claim 19, further comprising:

defining a plurality of composite flow items each comprising a specified user interface page; and

defining relationships between said plurality of composite flow items, wherein at least one composite flow item is a source flow item, and said specified user interface page is a specified source user interface page.

29. (Previously Presented) A method according to claim 27, wherein at least one specified user interface page is a composite user interface page.

30. (Previously Presented) A method according to claim 20, wherein at least one specified user interface page is a composite user interface page, the method further comprising:

modelling manipulations which are applied to said at least one page element within a specified source user interface page to create said composite user interface page.

31. (Original) A method according to claim 30, further comprising:

specifying an ordered plurality of manipulations to be carried out to create said composite user interface page.

32. (Previously Presented) A method according to claim 17, further comprising modelling at least one further user interface element to be included in the composite user interface.

33. (Previously Presented) A method according to claim 17, further comprising: processing said model to generate rules for communication between said composite user interface and said at least one source application.

34. (Original) A method according to claim 33, further comprising:

storing said rules within a hierarchical data structure comprising a plurality of entities.

35. (Original) A method according to claim 34, further comprising:

storing within said hierarchical data structure an entity representing the composite user interface; and

associating with said entity a data group providing configuration data for the composite user interface.

36. (Original) A method according to claim 35, further comprising:

storing within said hierarchical data structure details of a plurality of service entities representing processing modules which are together adapted to process user requests input to said composite user interface to produce one or more requests to at least one source application.

37. (Original) A method according to claim 36, wherein at least some of said service entities have an associated data group storing configuration data.

38. (Original) A method according to claim 37, wherein one of said service entities is an aggregation service entity representing an aggregation service configured to generate source application requests from a user request.

39. (Original) A method according to claim 38, wherein said aggregation service entity comprises:

a child entity representing the composite user interface; and said child entity has at least one child entity representing a source application.

40. (Previously Presented) A method according to claim 34, wherein said rules are generated using a plurality of writers each writer being associated with an entity in said hierarchical data structure, and being adapted to write data to a data group associated with the respective entity.

41. (Original) A method according to claim 40, wherein processing said model comprises:

selecting one or more objects within said model;

determining one or more writers to be invoked to write data from the or each object to data groups in said hierarchical data structure; and

invoking the or each writer to write data to said hierarchical data structure.

42. (Original) A method according to claim 41, further comprising: determining from said at least one writer at least one further object within said model, and processing said further object.

43. (Previously Presented) A method according to claim 41, further comprising:

identifying a further writer configured to identify an entity within said hierarchical data structure to which data is to be written.

44. (Original) A method according to claim 43, wherein identifying an entity comprises:

attempting to locate an entity within said hierarchical data structure to which data should be written; and

if said attempt is unsuccessful, creating an appropriate entity.

45. (Previously Presented) A method according to claim 40, wherein each writer is a writer object which is an instance of a respective writer class.

46. (Original) A method according to claim 45, wherein each writer class has a corresponding writer factory class.

47. (Original) A method according to claim 46, further comprising:

registering each writer factory class with a writer lookup object;

providing details of the or each object to be processed to said writer lookup object ; and

identifying one or more factory classes which should be used to create writer objects.

48. (Previously Presented) A non-transitory program memory comprising computer program code stored therein to cause a computer to carry out a method according to claim 1.

49. (Previously Presented) A computer apparatus comprising: a program memory that includes processor readable instructions; and

a processor for reading and executing the instructions contained in the program memory;

wherein said processor readable instructions comprise instructions controlling the processor to carry out the method of claim 1.

50. (Currently Amended) Apparatus adapted to generate model data representing a model of a composite user interface comprising a plurality of user interface elements provided by ~~at least one~~ a plurality of source ~~application~~ applications, the apparatus comprising:

a program memory that includes processor readable instructions;

a processor for reading and executing the instructions contained in the program memory, wherein the instructions are for:

generating model data representing a model of at least part of a user interface provided by ~~the or~~ each of the source ~~application~~ applications; and

generating model data representing a model of relationships between the at least part of the user ~~interface~~ interfaces provided by the source ~~application~~ applications and the composite user interface.

51. (Previously Presented) Apparatus according to claim 50, wherein said instructions for generating model data comprise instructions to:

define a plurality of source flow items each comprising a specified source

user interface page provided by a source application; and

define relationships between said plurality of source flow items.

52. (Previously Presented) Apparatus according to claim 51, wherein said instructions for generating modeldata representing a model of at least part of a user interface comprise instructions for defining at least one page element within each specified source user interface page.

53. (Previously Presented) Apparatus according to claim 51, wherein said instructions for generating model data representing a model of at least part of a user interface comprise instructions for defining at least one flow control condition, and means for associating a flow control condition with at least one of said plurality of source flows.

54. (Previously Presented) Apparatus according to claim 51, wherein said instructions for generating model data representing a model of at least part of a user interface comprise instructions defining request parameters used to obtain each specified source user interface page.

55. (Previously Presented) Apparatus according to claim 51, w wherein said instructions for generating model data representing a model of at least part of a user interface comprise instructions for defining at least one rule for each specified source page which can be applied to enable recognition of the associated specified source page.

56. (Original) Apparatus according to claim 55, wherein said at least one rule is specified using a regular expression, or a path expression.

57. (Previously Presented) Apparatus according to claim 50, wherein said instructions for generating model data representing a model of at least part of a user interface comprise instructions for creating a plurality of objects which are instances of classes defined in an object oriented programming language.

58. (Previously Presented) Apparatus according to claim 50, wherein said instructions for generating model data representing a model of relationships comprise instructions combining means for combining at least part of a plurality of source application models.

59. (Previously Presented) Apparatus according to claim 50, further comprising instructions executable by the processor for defining a plurality of composite flow items each comprising a specified user interface page, and defining means for defining relationships between said plurality of composite flow items.

60. (Previously Presented) Apparatus according to claim 51, further comprising instructions executable by the processor for defining a plurality of composite flow items each comprising a specified user interface page, and instructions executable by the processor for defining relationships between said plurality of composite flow items, wherein at least one composite flow item is a source flow item, and said specified user interface page is a specified source user interface page.

61. (Previously Presented) Apparatus according to claim 59, wherein at least one specified user interface page is a composite user interface page.

62. (Previously Presented) Apparatus according to claim 52, wherein at least one specified user interface page is a composite user interface page, the apparatus further comprising means for modelling manipulations which are applied to said at least one page element within a specified source interface page to create said composite user interface page.

63. (Previously Presented) Apparatus according to claim 62, further comprising instructions executable by the processor for specifying an ordered plurality of manipulations to be carried out to create said composite user interface page.

64. (Previously Presented) Apparatus according to claim 50, further comprising modelling at least one further user interface element to be included in the composite user interface.

65. (Previously Presented) Apparatus according claim 50, further comprising instructions executable by the processor adapted to process said model to generate rules for communication between said composite user interface and said at least one source application.

66. (Previously Presented) Apparatus according to claim 65, further comprising instructions executable by the processor for storing said rules, said instructions executable by the processor being adapted to store said rules in a hierarchical data structure comprising a plurality of entities.

67. (Previously Presented) Apparatus according to claim 66, wherein said instructions executable by the processor are adapted to store in said hierarchical data structure an entity representing the composite user interface and to store a data group providing configuration data for the composite user interface.

68. (Previously Presented) Apparatus according to claim 67, wherein said instructions executable by the processor are adapted to store within said hierarchical data structure details of a plurality of service entities representing processing modules which are together adapted to process user requests input to said composite user interface to produce one or more requests to at least one source application.

69. (Original) Apparatus according to claim 68, wherein at least some of said service entities have an associated data group storing configuration data.

70. (Original) Apparatus according to claim 69, wherein one of said service entities is an aggregation service entity representing an aggregation service configured to generate source application requests from a user request.

71. (Original) Apparatus according to claim 70, wherein said aggregation service entity comprises:

a child entity representing the composite user interface; and

said child entity has at least one child entity representing a source application.

72. (Previously Presented) Apparatus according to claim 66, comprising instructions executable by the processor to write said rules, each writer being associated with an entity in said hierarchical data structure, and being adapted to write data to a data group associated with the respective entity.

73. (Previously Presented) Apparatus according to claim 72 comprising instructions executable by the processor:

for selecting one or more objects within said model;

for determining one or more writers to be invoked to write data from the or each object to data groups in said hierarchical data structure; and

for invoking the or each writer to write data to said hierarchical data structure.

74. (Previously Presented) Apparatus according to claim 73, further comprising instructions executable by the processor:

for determining from said at least one writer at least one further object within said model; and

for processing said further object.

75. (Previously Presented) Apparatus according to claim 73, further instructions executable by the processor for identifying a further writer configured to identify an entity within said hierarchical data structure to which data is to be written.

76. (Previously Presented) Apparatus according to claim 75, further comprising instructions executable by the processor:

to attempt to locate an entity within said hierarchical data structure to

which data should be written;

to determine whether said attempt is successful or unsuccessful; and

for creating an appropriate entity if said analysis means determines that said attempt is unsuccessful.

77. (Previously Presented) Apparatus according to claim 72, wherein the instructions comprise a writer object which is an instance of a respective writer class.

78. (Previously Presented) Apparatus according to claim 77, wherein the instructions comprise a writer factory class associated with each writer class.

79. (Previously Presented) Apparatus according to claim 78, further comprising instructions executable by the processor::

for registering each writer factory class with a writer lookup object;

for providing details of the or each object to be processed to said writer lookup object; and

for identifying one or more factory classes which should be used to create writer objects.

80. (Previously Presented) A server configured to provide at least one composite user interface to a plurality of source applications, the composite user interface comprising a plurality of user interface elements provided by the source applications,, the server comprising:

a processor; and

a program memory, coupled to the processor, that includes processor readable instructions for:

combining a subset of the plurality of user interface elements from the source applications into composite user interface data;

providing the composite user interface data to a user computer system for display as the composite user interface by the user computer system, wherein the composite user interface causes the user

computer system to display the plurality of user interface elements provided by the source applications;

receiving a user request from the user computer system relevant to at least one of the source applications;

processing a model representing said composite user interface for communication between said composite user interface and the source applications; and

generating one or more source requests to each relevant source application that represents the user request.

81. (Original) A server according to claim 80, wherein said model comprises a model of at least part of a user interface provided by the or each source application and a model of relationships between the at least part of the user interface provided the or each source application and the composite user interface.

82. (Previously Presented) A server according to claim 80, further comprising storage means storing a hierarchical data structure comprising a plurality of entities, said hierarchical data structure storing said rules.

83. (Original) A server according to claim 82, wherein said hierarchical data structure comprises an entity representing the composite user interface, and said entity is associated with a data group providing configuration data for the composite user interface.

84. (Original) A server according to claim 83, wherein said hierarchical data structure comprises a plurality of service entities representing processing modules which are together adapted to process user requests input to said composite user interface to produce one or more requests to at least one source application.

85. (Original) A server according to claim 84, wherein at least some of said service entities have an associated data group storing configuration data.

86. (Original) A server according to claim 85, wherein one of said service entities is an aggregation service entity representing an aggregation service configured to generate source application requests from a user request.

87. (Original) A server according to claim 86, wherein said aggregation service entity comprises:

a child entity representing the composite user interface; and

said child entity has at least one child entity representing a source application.

88. (Previously Presented) A server according to claim 82, further comprising a plurality of writers adapted to generate said rules, each writer being associated with an entity in said hierarchical data structure, and being adapted to write data to a data group associated with the respective entity.

89. (Original) A server according to claim 88, wherein said processor comprises:

selecting means for selecting one or more objects within said model;

determining means for determining one or more writers to be invoked to write data from the or each object to said hierarchical data structure; and

means for invoking the or each writer to write data to said hierarchical data structure.

90. (Original) A server according to claim 89, further comprising means for determining using said at least one writer at least one further object within said model, and processing said further object.

91. (Previously Presented) A server according to claim 89, further comprising a further writer comprising identifying means configured to identify an entity within said hierarchical data structure to which data is to be written.

92. (Original) A server according to claim 91, wherein said identifying means comprises:

means for attempting to locate an entity within said hierarchical data structure to which data should be written; and

means for creating an appropriate entity if said attempt is unsuccessful.

93. (Previously Presented) A server according to claim 88, wherein each writer is a writer object which is an instance of a respective writer class.

94. (Original) A server according to claim 93, wherein each writer class has a corresponding writer factory class.

95. (Original) A server according to claim 94, further comprising:

a writer lookup object, said writer lookup object comprising means adapted to register each writer factory class with said writer lookup object,

means for providing details of the or each object to be processed to said writer lookup object ; and

means for identifying one or more factory classes which should be used to create writer objects.

96. (Currently Amended) A computer apparatus for generating a composite user interface for communication with a plurality of source applications, the apparatus comprising:

a program memory that includes processor readable instructions;

a processor for reading and executing the instructions contained in the program memory, wherein the instructions are for:

generating model data representing a model of said composite user interface in response to user input, wherein the composite user interface comprises a plurality of user interface elements provided by the source applications;

storing said model;

reading said model from said memory and generating a configuration data structure;

receiving a request from a composite user interface;

generating a source application request to at least one of said source application in response to said request, in accordance with data stored in said configuration data structure; and

transmitting said source application request to said at least one of said source applications.

97. (Currently Amended) A method for modelling and generating a composite user interface comprising user interface elements provided by ~~at least one~~ a plurality of source ~~application~~ applications comprising:

generating a source application model for each of the ~~at least one~~ plurality of source applications;

generating a composite application model using ~~the or~~ each source application model; and

processing said composite application model to generate rules for communication between said composite application and the source ~~application~~ applications.

98. (Currently Amended) A method for providing a composite user interface comprising a plurality of user interface elements provided by at least one source application, the method comprising:

monitoring operation of the composite user interface to obtain management data, wherein the management data includes usage conditions, wherein the usage conditions include demand for information and response time for providing the demanded information; and

modifying said composite user interface by changing a number of the user interface elements for display by a user computer system in accordance with the usage conditions.

99. (Previously Presented) A method according to claim 98, wherein the user interface elements comprise mandatory and nonmandatory user interface elements, the method further comprising modifying said composite user interface in response to said management data to (i) display the nonmandatory user interface elements during a first usage condition and (ii) not display the mandatory user interface elements during a second usage condition, wherein the first usage condition represents a lower usage than the second usage condition.

100. (Original) A method according to claim 99, wherein said modifying comprises deleting some of said plurality of user interface elements from said composite user interface.

101. (Previously Presented) A method according to claim 98, further comprising producing data representing usage patterns of said composite user interface using said management data.

102. (Previously Presented) A method according to claim 98, further comprising:

receiving at least one request message generated by said composite user interface;

producing at least one further message in response to said request message; and

forwarding said at least one further message to one of said at least one source applications.

103. (Previously Presented) A non-transitory program memory comprising computer program code stored therein to cause a computer to carry out a method according to claim 98.

104. (Previously Presented) A computer apparatus comprising:

a program memory containing processor readable instructions; and

a processor for reading and executing the instructions contained in the

program memory;

wherein said processor readable instructions comprise instructions controlling the processor to carry out the method of claim 98.

105. (Currently Amended) Apparatus for providing a composite user interface comprising a plurality of user interface elements provided by at least one source application, the apparatus comprising:

a program memory that includes processor readable instructions;

a processor for reading and executing the instructions contained in the program memory, wherein the instructions are for:

monitoring operation of the composite user interface to obtain management data, wherein the management data includes usage conditions, wherein the usage conditions include demand for information and response time for providing the demanded information; and

modifying said composite user interface by changing a number of the user interface elements for display by a user computer system in accordance with the usage conditions.

106. (Previously Presented) Apparatus according to claim 105, wherein the user interface elements comprise mandatory and nonmandatory user interface elements, further comprising modifying means adapted to modify said composite user interface in response to said management data to (i) display the nonmandatory user interface elements during a first usage condition and (ii) not display the mandatory user interface elements during a second usage condition, wherein the first usage condition represents a lower usage than the second usage condition.

107. (Previously Presented) Apparatus according to claim 106, wherein said instructions are further executable by the processor for deleting some of said plurality of user interface elements from said composite user interface.

108. (Previously Presented) Apparatus according to claim 105, further comprising instructions executable by the processor for producing data representing usage patterns of said composite user interface using said management data.

109. (Previously Presented) Apparatus according to claim 105, further comprising instructions executable by the processor for:

receiving at least one request message generated by said composite user interface;

producing at least one further message is response to said request message; and

forwarding said at least one further message to one of said at least one source applications.

110. (Previously Presented) A method for generating a composite user interface comprising a plurality of user interface elements provided by at least one source application, the method comprising:

combining a subset of the plurality of user interface elements from the source applications into composite user interface data; and

selecting said composite user interface from a plurality of predefined composite user interfaces on the basis of at least one predefined parameter.

111. (Original) A method according to claim 110, wherein said at least one predefined parameter comprises a parameter relating to at least one of time of day and date.

112. (Previously Presented) A method according to claim 110, wherein said at least one predefined parameter comprises a parameter relating to usage statistics of said composite user interface.

113. (Previously Presented) A method according to claim 110, wherein said at least one predefined parameter comprises a parameter relating to a marketing campaign operated by a business enterprise using said composite user interface.

114. (Previously Presented) A method according to claim 110, further comprising:

receiving at least one request message generated by said composite user interface;

producing at least one further message in response to said request message; and

forwarding said at least one further message to one of said at least one source applications.

115. (Original) A method according to claim 114, further comprising:

storing details of a plurality of user interface elements which are expected in response to said at least one further message;

storing data indicating that at least one of said expected user interface elements is mandatory, and at least one of said expected user interface elements is non-mandatory

receiving a plurality of user interface elements in response to said at least one further message;

generating a part of said composite user interface when all mandatory user interface elements have been received.

116. (Original) A method according to claim 115, wherein at least two of said plurality of predefined composite user interfaces comprise identical source user interface elements.

117. (Original) A method according to claim 116, wherein said at least two of said plurality of predefined composite user interfaces have different mandatory user interface elements.

118. (Previously Presented) A method according to claim 110, wherein at least two of said plurality of predefined composite user interfaces comprise different source user interface elements.

119. (Previously Presented) Apparatus for generating a composite user interface comprising a plurality of user interface elements provided by at least one source application, the apparatus comprising:

a program memory that includes processor readable instructions;

a processor for reading and executing the instructions contained in the program memory, wherein the instructions are for:

combining a subset of the plurality of user interface elements from the source applications into composite user interface data;

storing a plurality of predefined composite user interfaces; and

selecting said composite user interface from one of said plurality of predefined composite user interfaces on the basis of at least one predefined parameter.

120. (Original) Apparatus according to claim 119, wherein said at least one predefined parameter comprises a parameter relating to at least one of time of day and date.

121. (Previously Presented) Apparatus according to claim 119, wherein said at least one predefined parameter comprises a parameter relating to usage statistics of said composite user interface.

122. (Previously Presented) Apparatus according to claim 119, wherein said at least one predefined parameter comprises a parameter relating to a marketing campaign operated by a business enterprise using said composite user interface.

123. (Previously Presented) Apparatus according to claim 119, further comprising instructions executable by the process for:

receiving at least one request message generated by said composite interface;

producing at least one further message is response to said request message; and

forwarding said at least one further message to one of said at least one source application.

124. (Previously Presented) Apparatus according to claim 123, further comprising instructions executable by the process for:

storing details of a plurality of user interface elements which are expected in response to said at least one further message;

storing data indicating that at least one of said expected user interface elements is mandatory, and at least one of said expected user interface elements is non-mandatory;

receiving a plurality of user interface elements in response to said at least one further message;

generating a part of said composite user interface when all mandatory user interface elements have been received.

125. (Original) Apparatus according to claim 124, wherein at least two of said plurality of predefined composite user interface comprise identical source user interface elements.

126. (Original) Apparatus according to claim 125, wherein said at least two of said plurality of predefined composite user interfaces have different mandatory user interface elements.

127. (Previously Presented) Apparatus according to claim 119, wherein at least two of said plurality of predefined composite user interface comprise different source user interface elements.

128. (Previously Presented) A non-transitory program memory comprising computer program code stored therein to cause a computer to carry out a method according to claim 110.

129. (Previously Presented) A computer apparatus comprising:

a program memory containing processor readable instructions; and

a processor for reading and executing the instructions contained in the program memory;

wherein said processor readable instructions comprise instructions controlling the processor to carry out the method of claim 110.